Moving Object Detection using Adaptive Blind Update and RGB-D Camera

Navid Dorudian; Stanislao Lauria; Stephen Swift

***Abstract*— A novel background subtraction approach using RGB-D camera and an adaptive blind updating policy is introduced. This method in initialization creates a model to store background pixels to compare each pixel of the new frame with the model in the same location to identify background pixels. The background-model update presented in this paper uses regular and blind update which also has a different criteria from existing methods. In particular, blind update frequently changes based on the background changes and the speed of moving object. This will allow the scene model to adapt to the changes in the background, detecting the stationary moving object and reducing the ghost phenomenon. In addition, proposed bootstrapping segmentation and shadow detection are added to the system to improve the accuracy of the algorithm in shadow and depth camouflage scenarios.**

The proposed method is compared with the original method and other state of the art algorithms. Experimental results show significant improvement in those videos that stationary object appear. In addition, the benchmark results also indicate strong and stable results compared to the other state of the art algorithms.

***Index Terms*— GPS-denied Environments, Dynamic Environments, Object Detection, Nonparametric Background Subtraction, Background-model Update, Segmentation**

I. Introduction

In the past few years background subtraction techniques have shown a growing interest among researchers in video surveillance applications [1]. Typically, the main goal of these applications are to separate the moving object known as foreground from other objects in the scene called background [2]. The creation of a background model is an important step of most background subtraction techniques since the background is the only reliable source of existence compared to foreground. One of the most common approaches is to compare the current image with previous images which are known as "reference" in the literature. Generally these references are created from a single image or more compound model which is known as a "scene model" [3]. Traditionally a scene model requires a regular update to adapt to the changes over the time in the real-world scenarios. These changes could be appearing as a new object in the scene, moving an object in the background or illumination changes. Also the type of samples we can use in the scene model and for how long it is valid, has always been discussed in the literature, for example first in first out is one of the classical approaches for updating the background model which discards the old samples in the model and substitutes them with the new pixels after several frames or seconds. These classical approaches update all the old pixels in the model which often is not essential as those pixels may still be valid samples. On the other hand, updating the scene model only by those pixels that are recognised as background or also involving the foreground pixels has always been discussed. These procedures are known as a blind and conservative update policy [4].

The conservative update scheme never incorporates those pixels classified as part of foreground region. Theoretically, this policy is a suitable choice which can produce a sharp detection of the moving objects. However, in most practical scenarios it can lead to deadlock situations and production of ghost phenomenon. For instance, a change to the background of the scene can cause the background model to incorrectly contain foreground samples. This prevents an update to the background model and therefore causes a permanent misclassification.

On the other side, blind update such as the method used in ViBe [4] and MoG [5] incorporates all sample values into the background model update regardless of being identified as foreground or background. This approach has some disadvantages such as the weak detection of slow-moving object or those foregrounds which stop and remain static for some period of time which these regions gradually will become part of the scene model. In the literature, these motionless objects are typically called stationary foreground objects (SFOs) [2]. Detection of SFOs is one of the well-known topics of background subtraction technique which attracted the attention of many researchers in the last few years[6]. In SFO scenarios, pixels of stationary object gradually absorb to the background model and eventually the model will adapt to the motionless object. In this paper, we have proposed a blind update scheme that is able to improve the detection accuracy of SFOs as well as fast moving objects.

II. Related Work

Recently a new nonparametric method called visual background extractor (ViBe) proposed by Barnich and Van

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2019.2920515, IEEE Sensors Journal

[4]. This method has been widely used by many researchers for having simple modelling, great accuracy and real-time processing. A background model will be created for each pixel by recently observed pixel values in the same location. This method starts the initialization with a single frame and gradually adapt to the environments. The foreground mask is made by simple comparison between existing pixel value and its background model in the same location. To adapt to the changes occurring in the scene, the background model uses two updating policies which known as random neighbourhood and conservative updating. Despite all the great results ViBe algorithm has achieved, it still suffers from the detection of stationary objects and production of ghost. Some methods have been introduced to solve this problem by using scene model of larger size. However, the drawback of such a system is that it requires a high memory usage or time. Therefore, the original ViBe algorithm has been modified in [7] to be able to detect slow moving objects without appearing as ghost. This method which called VIBeF, foreground model with adaptive approach is proposed to support the ViBe with blind update. The authors of ViBeF recommended this method can adapt to complex scenarios including illumination changes and variations in background objects.

The authors of [6] proposed a multi-object tracker adapted for conveying systems which is based on a feedback loop from tracking to detection. The main goal of their work was to stop the adaptation of the regions which belongs to stationary object. They have used state of the art background subtraction techniques ViBe [4] and the gaussian mixture model (GMM) [8] to implement their ideas. The neighboring update process of ViBe has been disabled for pixels of stationary objects to prevent the absorption of SFOs into the background model. On the other hand, the learning rate of α is considered to incorporates pixel samples in the GMM model. For the stationary objects, the α is set to zero. According to the authors, significant improvements of tracking results has been achieved in real video sequences.

The new method proposed in [9] efficiently identifies SFOs based on three nonparametric background models (long term, medium term and short term). The goal of this method was to improve the detection quality of classical moving object detection and using novel Finite State Machine in scenarios featuring moving objects that becomes motionless (e.g. people in offices or vehicles on urban roads).

Using color and depth data in nonparametric methods such as ViBe to detect moving object is not new and this approach has previously been applied in [10][11][12][1]. However, the vision system that the authors used in [10] and [11] consist of a separate TOF (Time of flight) sensor and RGB camera. Some other authors such as [12] tried a standard stereo camera. Recently, a new approach proposed in [1] which incorporates innovative mechanisms to detect micro air vehicles (MAVs) in GPS-denied environments using an external RGB-D sensor. This method stores several color and depth frames as a model and then compares individual pixel from a frame with the stored models to identify the pixel as a foreground or background. Additionally, regular update and blind update is used to adapt the model to the changes in the background. Despite all the effort made for this method to detect the moving object in dynamic background and other challenging scenarios, it still suffers low detection accuracy in stationary moving object as the foreground progressively absorbs to the background model by blind update.

III. THE PROPOSED ALGORITHMS

In the last few years low cost RGB-D sensors such as Microsoft Kinect attracted many researchers to detect and recognize objects and behaviors such as motion [13], Suicide [14] ,drone [1] and obstacle detection [15]. These devices are able to produce well calibrated RGB and depth frames and normally capture 30 frames per second which is appropriate for motion detection algorithms.

The proposed method in this paper is an improved method introduced in [1]. The main novelties and contributions of this research are as following; 1) Adding adaptive blind update method which changes the frequently of blind update based on the speed of moving object. 2) The segmentation rules are more effective and complex compared to the original method. 3) Bootstrapping detection and segmentation proposed to improve the accuracy in bootstrapping scenarios. Shadow detection method proposed based on L*a*b* color space. The method is also fully evaluated in all sequences of SBM-RGBD challenge dataset [24]. Bootstrapping sequences are defined as those sequences that foreground objects exist in all frames from the beginning[16].

Our method creates background models by history of previously observed pixel values. Then current pixels will be compared to the model for foreground segmentation. The proposed method involves different phases to cope with the changes in the background and effectively works in live application. Figure 1 demonstrate the flow chart of the proposed method.

This method produces one model for color images and another model for depth images. The system stores the first N (N = 20 in our experiments) number of frames to initialise the models (system initialization step). Typically, depth frames are noisy and have some limitations for certain materials, surfaces and black color which known as “holes” [17] or “Absent Depth Observations (ADO) ” [3]. In the proposed method these unknown pixel values in the depth frame will be filled by neighboring values before storing in the model. This process is called “ADO removal filter”.

The segmentation process starts after the minimum number of samples stored in the model. If any moving object has been detected during the initialization stage, then bootstrapping is occurred. With Fig. 1. The far-right box has been clipped, we called this stage “Check Bootstrapping”.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2019.2920515, IEEE Sensors Journal

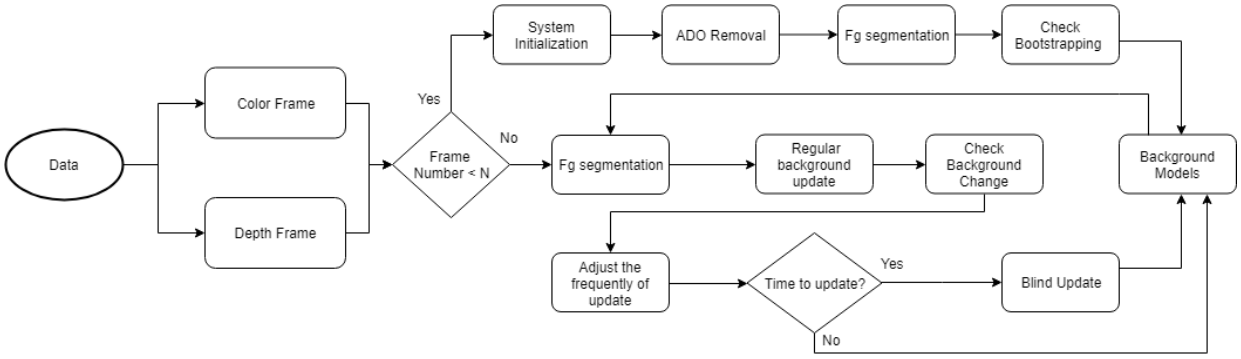


Fig. 1. Flow chart of the proposed object detection method.

When bootstrapping is occurring, some pixels of foreground stores in the background model. Therefore, the segmentation result (foreground detection) will be only a fraction of foreground. To solve this problem, during initialization stage we attempt to find the shape of each object existing in the scene by using edge detection techniques. This has been reached by combining the results of color and depth frames achieved by canny edges detection algorithm [18]. The system checks each object exists in the scene with the segmentation result (foreground). If a high percentage of any object in the scene identified as a foreground, then the whole object will be added to the foreground mask. This significantly aids the system to increase the detection rate in the bootstrapping scenarios. However, in some rare cases this could also increase misclassification.

The background model will be completed after initialization and the system goes to the main loop to start by comparing individual pixels to the model to classify segmentation results which is called "Fg segmentation".

After segmentation, the background model will be updated with two methods of "regular background update" and "blind update". First only those pixels marked as a background will be updated. In addition, after α number of frames the system blindly swaps the new pixels with background model regardless of being marked as foreground or background. The frequently of blind update (α) will be changed based on the speed of moving object.

On the other hand, in "check the background change" when a change in the background is occurring, the system will significantly increase the frequently of blind update in order the model adapt to the changes in the background. The system identifies a change in the background by comparing depth frame with a sample depth image (usually with initial frame when the moving object doesn't exist in the scene). If an area of a scene has a longer depth compare to the samples, it demonstrates a change in the background has occurred.

As figure 1 demonstrates, the proposed method is based on three main steps of initialization, segmentation and update. In the remaining of this section the key steps of Figure 1 will be discussed in more detail.

A. Initialization

Background subtraction methods typically require a scene model to compare and segment the regions of the frames as a background or foreground. In the meantime, every model needs a size and initialization procedure which significantly has an effect on speed and accuracy of the method. Higher amount of sample makes the method more accurate, but it also causes the method to be slower and take longer to start segmentation. Therefore, if the system runs on a real-time the size of model should be chosen based on the hardware.

The proposed method for full system initialization requires N (N=20 in our experiments) number of frames to produce the complete model. However, the system starts segmentation when it passes *MinFrame* (*MinFrame*=3) which is the minimum number of required samples to identify foreground.

Most of other methods require a high number of frame and time for initialization. However, in comparison to the other methods, our approach can complete initialization at a faster time. This will be carried out by blindly storing the first N number of depth and color frames in to the model and then with the aid of the updated stage the model will gradually improve. This process will help the system to start rapid detection of moving object.

B. Bg/Fg Segmentation

Traditionally Probability Density Functions (PDF) and statistical parameters such as mean or variance are the most common components of the background subtraction algorithms. Alternatively, statistical significance can be used to build a model based on previously observed color pixel values and depth data. The hypothesis is that, if the same pixel value has been observed number of times in the same location, the pixel has a high chance of being background.

The process of background subtraction will require classification of each pixel as background or foreground. The value of current pixel in color frame will be compared with the color model in each location to discover if it is close to some of the samples in the model. In parallel, depth pixels will be compared to depth model to determine if the pixel is in the same distance or further to the camera.

In some cases, RGB and depth have the same individual

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2019.2920515, IEEE Sensors Journal

segmentation result. In other words, by comparing the pixel to the models, both (color and depth) individually agree whether the pixel is part of the background or not. However, in some other challenging scenarios they are strongly against each other. This means one of the sensors (color or depth) has been affected by the noise or limitation of the sensor. An example of these situations could be color camouflage such as foreground having the same color as the background, change of illumination, shadow or depth camouflage such as moving the hand on the wall.

In order to produce the final decision (foreground mask), the system should find out from which sensor the noise is coming from and which sensor is more reliable. We have introduced a set of rules based on facts that the system follows in segmentation process to reduce the effect of these noises for producing the foreground masks. Figure 2 demonstrate the rules which the system follows for classification of each pixel.

Unlike RGB cameras, depth images are resistant to illumination effects. In addition, in the last few years the depth accuracy has been significantly increased with the creation of new sensors such as Microsoft Kinect V2 sensor [19][20]. Consequently, our method has relied more on depth outcome to produce the result.

To improve the accuracy of our method, we have used CIE L*a*b* color coordinate for color images which is defined by Hunter and Harold in 1987[21]. One of the most significant characteristics of the L*a*b* space is device independence. L*a*b* color space is built on one channel for Luminance (lightness) (*L*) and two other channels for color (green–red and blue–yellow) (*a and b*). $L^* = 0$ represents the darkest black, and $L = 100$ the brightest white. a^* and b^*, will represent true neutral grey values at $a^* = 0$ and $b^* = 0$. The a^* axis represents green at negative values and red at positive values. The b^* axis represents blue at negative values and yellow at positive values.

Unlike the RGB space, L*a*b* color is intended to approximate human vision where the *L* element closely indicates human perception of lightness. Thus, it can be used to check the color value of pixels (*a and b*) without interfering of illumination (*L*) component. Using L*a*b* color space significantly helped us to improve the segmentation accuracy and detection of shadow area. Our method identifies an area as a shadow where the *L* component is low (close to 0) and the depth frame on that location shows no change compare to the depth model.

Formally, if we denote a 3d point as a $X=(x,y,z) \in R^3$, *d(X)* the value in the depth and *v(X)* the value in a given color at location *X* in the new frames. $v_i$and $\mathrm{d_i}$ shows an index of *i* in a background sample value of each background pixel located at *X* which demonstrated by a collection of *N* background depth and color sample values taken before as:

$$M(X)_{Lab}=\{v_1,v_2,v_3,\ldots, v_n\} \quad (1)$$

$$M(X)_D=\{d_1,d_2,d_3,\ldots, d_n\} \quad (2)$$

We refer to fg as a foreground pixel, bg as a background pixel, $M(X)_D$ as a background depth model and $M(X)_{Lab}$ as a background color model at location *X*.

To identify each pixel of new frame as a bg or fg, the system checks the depth data. If the depth is ADO (unknown value) at that pixel location, it only compares the color value with the $M(X)_{Lab}$. This mean, the system will only rely on the decision of the color if depth value is not available in any pixel location. If the difference of color value is smaller than $\mathrm{Th_{RGB}}$(Acceptable threshold) we count the pixel as a similar color. Each pixel location which find at least cardinality amount denoted by $\#_{Min}$ similar pixels will be classified as bg. Although, the $\#_{\mathrm{Min}}$ is insensitive, it has positive correlation with number of samples. This means higher samples in the model require higher $\#_{\mathrm{Min}}$ value. Therefore, as a default we recommend $\#_{Min} = N/5$.

It then checks if bootstrapping has occurred in the sequence. If the system detects any moving object during the initialization stage, it will categorize the sequence as a bootstrapping. Normally when bootstrapping is occurring, the moving object (foreground) will exist in the background model. This will significantly reduce detection rate. To improve the detection in bootstrapping sequences we have set up a special rule.

We attempt to find the shape of each object existing in the scene by combining the results of color and depth canny edge detection. Then extracts all contours of the scene as an object. The system checks each object exists in the scene with the segmentation result (foreground). If a high percentage of any object in the scene identified as a foreground, then the whole object will be added to the foreground mask.

In the next step, we compare all the non-ADO depth pixel with $M(X)_D$. If the difference is larger than Th_D (Acceptable depth tolerance threshold), we count the pixel as having longer depth value. Each pixel location which find at least $\#_{Min}$ similar pixels, will be classified as bg. Then, pixels which are on the edges and around object boundaries will be only classified based on color result as depth value is not accurate around object boundaries [22][23]. All other pixels will be compared with the $M(X)_D$. Those pixels who cannot find $\#_{Min}$ in the same or longer range by considering some tolerance Th_D, will be classified as fg. for the remaining pixels, the color value of *L* component will be then checked. If it represents a dark value, then we count this area as a shadow and the pixel will be classified as bg. The remaining pixels will be compared with the depth model again without any tolerance this time, if it could find enough similarity, the pixel will be classified as bg. All other pixels will be decided by comparing to the $M(X)_{Lab}$.

Note the value of tolerance in this system is different with other methods. Experimentally we have realised that depth sensors are more accurate in closer areas. Therefore, the depth tolerance value should depend on the distance of the pixels to the sensor. In fact, the longer the pixel is, the higher the tolerance need to be.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2019.2920515, IEEE Sensors Journal

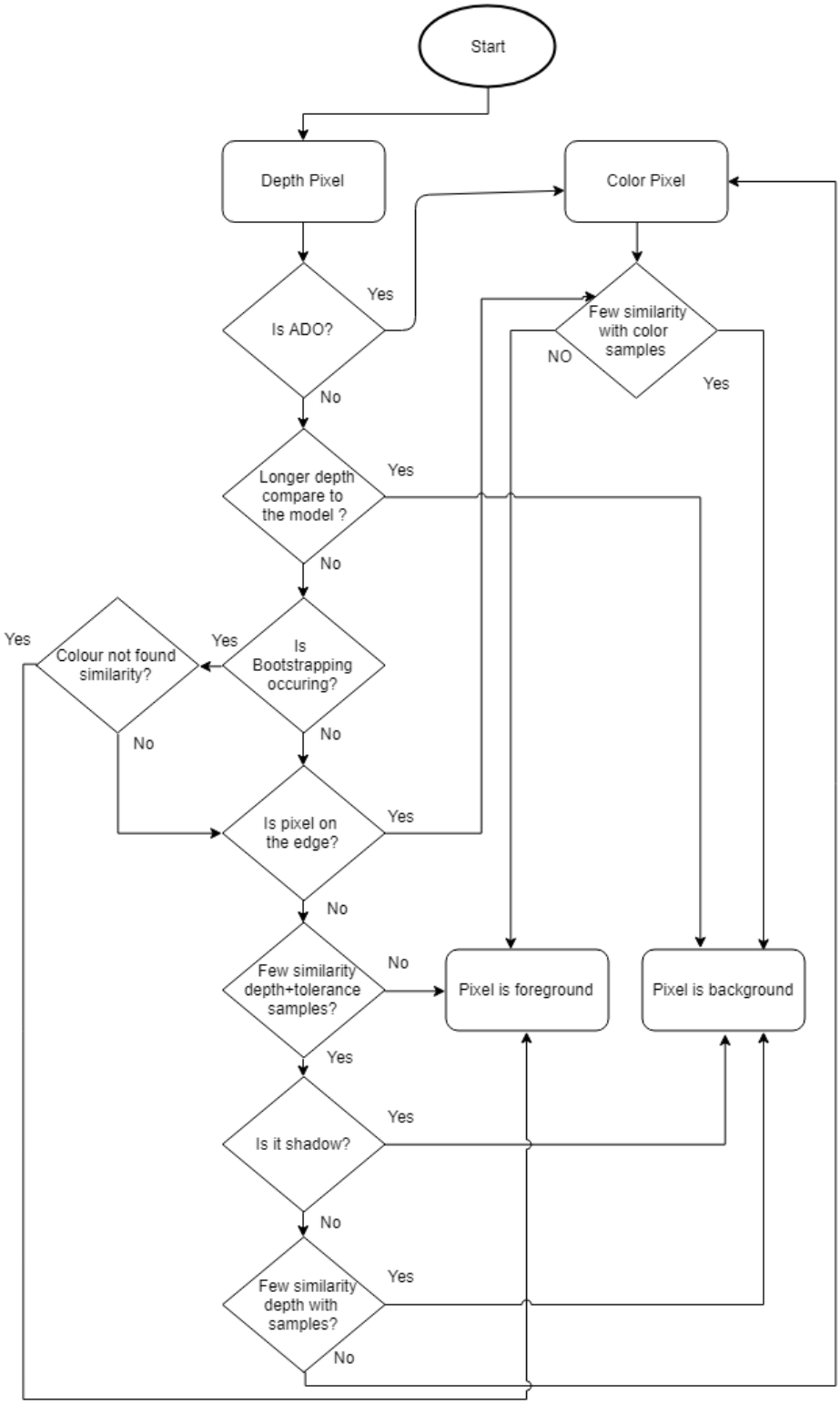


Fig. 2. Flow chart of the proposed object detection method.

C. Background Model Update

The proposed method is using two methods to continuously update the background model with the new frames. Regular and blind update aids the models to adapt to the changes in the background over the time. An example of these changes is a new object appearing in the scene, illumination changes and change to the location of an object completely in a different area to the background.

Regular update:

Similar to [1], those pixels identified as a background, will randomly swap the value of the pixel in the new frame with

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2019.2920515, IEEE Sensors Journal

one of the samples in color model in the same location. In the depth models, the system checks the distance (depth value) of the new pixel with background model. In particular, the system searches for the smallest sample in the depth model and compares it with the value of new depth frame in the same location. If the new frame has further distance, the model replaces the new value. The main drawback of this update is that if any background object moves forward, such as moving a table in the middle of the room, the system will never update this section and it will permanently misclassify this as a foreground. For this reason, blind updated stage added into our algorithm allows the system to adapt to the changes in the background over the time.

Blind update policy

Conservative scheme that has been used in regular update only effect those pixels that have been identified as part of background. The main drawback of this policy is that misclassified foreground pixels will be held in the background model. Then real background pixels will permanently classify as foreground and never enter the scene model. This will lead to deadlock situation and creation of ghost phenomenon. On the other side, blind policy updates the scene model with any pixel regardless of being as part of foreground or background. This policy has some advantages such as being able to adapt to the changes in the background and prevent the production of ghost phenomenon. The weakness of blind update is that slow and stationary objects will gradually become part of the background. Moreover, the frequently of blind update has been also discussed in the literature. Using high frequently of blind update could cause some foreground pixels misclassified as background. On the other hand, the background pixels are rarely misclassified.

In this paper, we are proposing an adaptive blind update for background model which reduce the weakness of blind update and allows the model to adapt to complex scenarios. The foreground in fast-moving objects are able to tolerate more frequent blind update than slow moving objects. This is because slow moving objects will gradually absorb to the model. Therefore, the frequently of blind update is very important and depends on the type of moving object (fast, slow or stationary). Consequently, by tracking the moving object, the frequency of blind update can be changed based on the speed of the object. We have defined three speeds categories for moving object as fast, slow and stationary. When the moving object is fast, the blind update could occur more often. In case of slow moving, we will reduce the frequently and once its stationary the blind update rate should be near zero. The proposed method is based on three main stages: 1) Constantly detection of moving object. 2) Track the moving object, 3) calculate the frequency of blind update.

In this research RGB-D sensor has been used which allows to produce the 3D position of any pixel in the scene. A simple tracking method has been used in this research to reduce the cost of computation although it is possible to employ more complex tracking methods.

If we denote the central position of the moving object as $X_t = (x_1, y_1, z_1)$ at the time *t*, a second after (t+1), the moving object will be at $X_{t+1} = (x_2, y_2, z_2)$. Then the distance taken by moving object can be calculate as:

$$D_m(X_t, X_{t+1}) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \qquad (3)$$

The distance taken by the foreground in a second can be used to change the frequency of blind update as the following:

$$\alpha = \begin{cases} High\ frequent\ update & D_m > d_{threshold} \\ Low\ frequent\ update & D_m < d_{threshold} \\ \approx 0 & D_m \approx 0 \end{cases} \qquad (4)$$

Where α is the frequently of blind update and $d_{threshold}$ is the amount of distance that define the fast or slow-moving object. If the moving object travel more than $d_{threshold}$ will be considered as fast moving and lower than that will be considered as slow-moving object.

IV. Results and Evaluation

To evaluate the performance of the proposed method, three experiments have been carried out in this paper. At first the proposed method has been compared in Total error (TE), Similarity measure (*S*) and Similarity measure in object boundaries (S_B) to the original method introduced in [1]. Then the result of proposed method has been compared with the other state of the art algorithm to measure the accuracy of our method. To achieve this the proposed method has been tested on the publicly available dataset called SBM-RGBD introduced in [24]. The ground-truth, ROI (region of interest) and a MATLAB code to evaluate the results has been provided with this dataset to compare the accuracy of individual method. In particular, the evaluation is based on following metrics: Recall, Specificity , False Positive Rate, False Negative Rate, Percentage of Wrong Classifications, Precision and F-Measure. This dataset has 33 sequences under seven different challenging categories in Illumination Changes, Color Camouflage, Depth Camouflage, Intermittent Motion, Out of Sensor Range, Shadows and Bootstrapping. Table 1 illustrates the summary of this dataset. We should state that the proposed method has been evaluated with the entire dataset and only one set of tuning parameters have been used to produce the segmentation result for all the sequences. At last we have tested the algorithm in the real-time to measure the computational cost of our method. This test shows that the system can successfully run the application real-time with approximately 12 fps (frame per second).

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2019.2920515, IEEE Sensors Journal

TABLE I
DETAILS OF SBM-RGBD DATASETS

Category	Sequence Name	Number of Frames	Number of Ground truth	Frame number where moving object occur	Main effected sensor	Test Objective
Bootstrapping	adl24cam0	70	4	1	Color Depth	Sequences containing foreground objects in all their frames
	bear_front	290	15	1		
	Bootstrapping_ds	399	11	1		
	fall01cam0	160	9	1		
	fall20cam0	110	6	1		
Color Camouflage	Cespatx_ds	429	11	133	Color	Sequences containing foreground objects that are having similar color to the background
	Hallway	618	18	48		
	colorCam1	300	29	57		
	colorCam2	360	26	61		
Depth Camouflage	DCamSeq1	600	46	1	Depth	Sequences containing foreground objects that are having similar depth to the background
	DCamSeq2	670	52	98		
	Despatx_ds	465	12	145		
	Wall	218	80	60		
Illumination Changes	ChairBox	529	62	1	Color	Sequences including strong and mild illumination
	Ls_ds	408	2	0		
	TimeOfDay_ds	1232	2	0		
	genSeq1	410	25	103		
Intermittent Motion	Shelves	554	134	183	Color Depth	Sequences with scenarios known for making “ghosting” artifacts in the detected motion. Removed foreground objects or abandoned foreground objects.
	Sleeping_ds	300	9	85		
	abandoned1	250	47	1		
	abandoned2	250	72	52		
	movedBackground1	250	78	1		
	movedBackground2	250	37	1		
OutOfRange	MultiPeople1	1190	39	118	Depth	Sequences including foreground or background objects that are too far or close from the sensor
	MultiPeople2	1400	53	91		
	TopViewLab1	670	33	110		
	TopViewLab2	650	33	120		
	TopViewLab3	700	39	135		
Shadows	Shadows_ds	331	9	150	Color Depth	Sequences that foreground objects caused creation of shadows. These are visible-light shadows in the color frames or IR shadows in the depth frames
	fall01cam1	160	7	62		
	genSeq2	300	26	119		
	shadows1	260	28	65		
	shadows2	250	26	104		

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2019.2920515, IEEE Sensors Journal

A. *Comparison to the original Method:*

In this section, the results achieved by the proposed method is compared with the original algorithm. For more precise comparison, four more sequences which stationary moving object occur in the scene from SBM-RGBD has been added to the publicly available sequences used in [1]. The following sequences has been used for comparison.

1. **SHSQ**: The aim of this sequence is to measure the effect of shadows in the scene by moving a box on the floor.
2. **Genseq:** This sequence has been made to measure the overall performance of the method by having several challenging scenarios that occur in one scene. This sequence is a scene with individual person moving a box containing strong and mild illumination changes.
3. **Colcam:** The moving object (the board) in this sequence has the same color as the background to explore the possible errors of the method in color camouflage.
4. **DCamSeq:** In this sequence moving object (the hand) is moving around the cupboard and therefore it has the same depth as the background to explore the possible errors of the method in depth camouflage.
5. **Abandoned1:** The scenarios in this video is made to test the tolerance of the method in creation of ghost phenomenon. A box will move from the stationary position to the top of the cupboard and stay stationary for some frames.
6. **Abandoned2:** This sequence is made to test the tolerance of the method for stationary foreground object. A bag will be dropped in the scene and remain there for a while.
7. **MovedBackground1:** It aims to measure the accuracy of the method in case of background changes. A box drops from a cupboard and stays on the floor.
8. **MovedBackground2:** The video starts with a bag on the floor and will be removed from the scene. Similar to moved Background1 the goal is to measure the algorithm in background changes.

To quantitatively evaluate the accuracy of each method, the following metrics has been used to compare the outcomes:

False Positive (FP): Pixels belong to the background which are classified as foreground.

False Negative (FN): Pixels belong to the foreground which are classified as background.

Total Error (TE): The total amount of misclassified foreground and background pixels which normalized to the image size.

Similarity measure (S): This non-linear metric previously has been used in [25] and called Jaccard's index [26]. It combines the FN and FP as the following:

$$S(A,B) = \frac{|A \cap B|}{|A \cup B|} \quad (5)$$

Where A denoted as identified foreground area and B is ground-truth. Higher result (close to 1) demonstrate foreground correctly detected similar to the ground-truth and lower result (Close to 0) shows smaller detection.

Similarity measure (S_B): It only explores the misclassified pixels surrounding the foreground objects boundaries. It is calculating as S, but only covers the regions of 10 pixels around the ground-truth boundaries.

An example of qualitative comparison is presented in figure 3 where (a) is color image, (b) depth image , (c) ground-truth, (d) original method and (e) proposed method. The sequences are dynamic and each row in the figure represents one of the last few frames of a sequence. These sequences are *abandoned1, moveBackground1, shseq, movedBackground2* and *abandoned2* sequences where stationary moving object exist. Segmentation results which are closer to the grand-truth (white area) show more accurate detection. The gray area shows outside of region of interest which is not part of evaluation. The proposed method clearly could achieve more accurate results in these stationary moving object sequences.

In the first sequence (abandoned1), the brown box moves from the floor to the top of cabinet and stay stationary. As the foreground stay stationary, it will gradually absorb to the model in the original method. However, in the proposed method the system significantly reduces the blind update as the item is stationary and consequently the foreground detection remains sharp and more accurate.

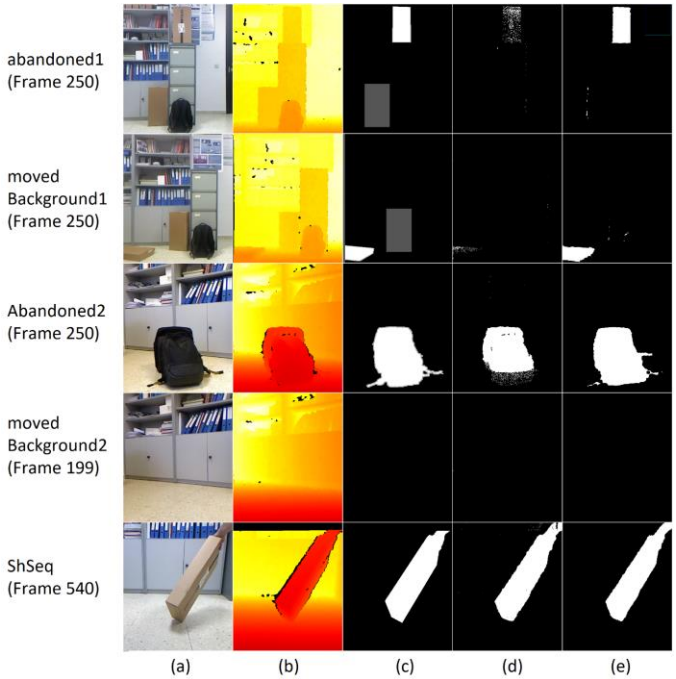


Fig. 3. (a) Color image, (b) Depth image, (c) Ground-truth, (d) Original method, (e) Proposed method

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2019.2920515, IEEE Sensors Journal

In moved background 1, the brown box drops from the top of cabinet to the floor and stay stationary there. The foreground gradually absorbs to the model due to the blind update. However, the proposed method can keep the sharp detection of the foreground due to the reduction of frequently in blind update. In abandoned 2 sequence, a black bag will be dropped in the scene and stay stationary there. The foreground starts absorbing to the model in the original method and the foreground gradually disappear in segmentation mask. This starts from the bottom of the bag as it is closer to the background. In moved background 2 the bag will be removed from the scene and both methods could successfully maintain the background model. In shseq both methods could avoid the shadow. However, due to the advanced segmentation method, the proposed algorithm could produce more accurate result in object boundaries, less noise and the hand is fully detected.

Figures 4,5 and 6 demonstrate the accuracy of the proposed method in this paper and original method (NBMS) [1] in TE, S and S_B. Lower amount of TE and higher S and S_B shows better performance. As illustrated in the figures, performance of both methods is similar to those sequences that the stationary object or slow-moving object doesn't exist for a while (*Genseq*, *Colcam*, *SHSQ*). However, in those sequences with stationary object (abandoned1, *abandoned2*, *movedBackground1*, *movedBackground2*) the proposed method could have achieved better results in all sequences. Figure 1 illustrate an example of stationary moving objects datasets and their results. The results show that both methods could completely prevent ghost production in *movedBackground2* sequence which the bag has been removed from the scene to measure the tolerance of the algorithms in background changes. However, in stationary moving objects the proposed method could achieve better results by preventing the absorption of the moving object into the background model.

B. SBM-RGBD Dataset:

In this section we have evaluated the proposed method with the entire SBM-RGBD datasets and compared our results with other state of the art methods in SBM-RGBD challenge. These methods are RGBD-SOBS[27], RGB-SOBS[28], SRPCA [16], SCAD[29], cwisardH+[30], AvgM-D, Kim and MFCN[31]. Table 2 demonstrate the average results of the entire dataset. A detailed result of proposed method (BSABU)

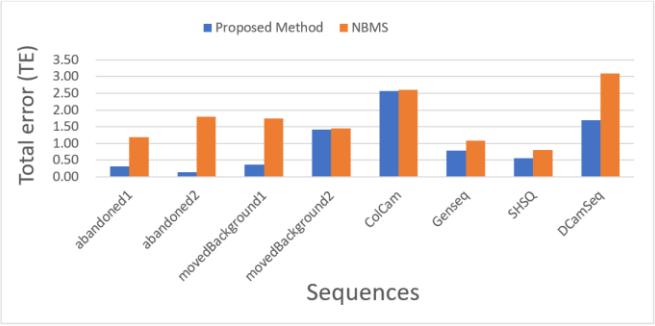


Fig. 4. Total error (TE) in different sequences. The lower amount shows better performance.

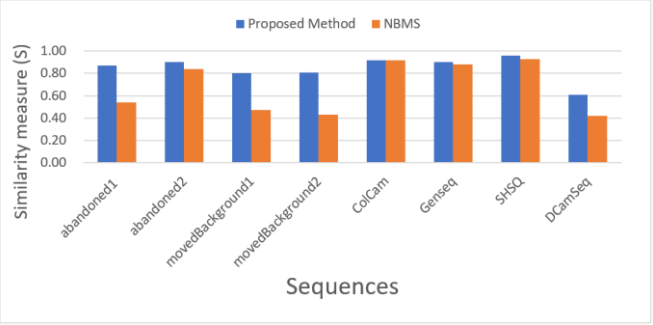


Fig.5.Similarity measure (S) in different sequences. The higher amount shows more similarity with the ground truth and therefore better

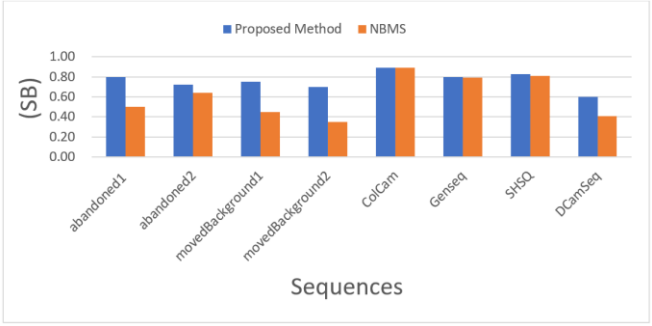


Fig. 6. Similarity measure in object boundaries (SB) in different sequences. The higher amount shows more similarity with the ground truth around the object boundaries and therefore better performance.

TABLE II
AVERAGE RESULTS OF ALL AVAILABLE METHODS IN SBM-RGBD DATASETS

Name	Recall	Specificity	FPR	FNR	PWC	Precision	F-Measure
RGBD-SOBS	0.8391	0.9958	0.0042	0.0895	1.0828	0.8796	0.8557
RGB-SOBS	0.7707	0.9708	0.0292	0.1578	5.4010	0.7247	0.7068
SRPCA	0.7787	0.9738	0.0262	0.1499	3.2243	0.7477	0.7474
SCAD	0.8847	0.9932	0.0068	0.0439	0.9088	0.8698	0.8757
cwisardH+	0.7622	0.9817	0.0183	0.1664	2.8806	0.7556	0.7470
AvgM-D	0.7065	0.9869	0.0131	0.2221	2.8848	0.7498	0.7157
Kim	0.8493	0.9947	0.0053	0.0793	1.0292	0.8764	0.8606
MFCN	0.9186	0.9984	0.0016	0.0100	0.2373	0.9103	0.9143
Proposed-Method (BSABU)	0.8211	0.9955	0.0045	0.1075	1.0854	0.8795	0.8477

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSEN.2019.2920515, IEEE Sensors Journal

for each sequence is available on[32].

The SBM-RGBD dataset comes with a MATLAB code to measure the accuracy of moving object detection across various challenges. Formally TP, TN, FP and FN show the total number of True Positive, True Negative, False Positive and False Negative for each video. The seven metrics used in this challenge for evaluating the results of moving object detection are:

1. *Recall* $$Rec = \frac{TP}{TP+FN} \quad (6)$$

2. *Specificity* $$Sp = \frac{TN}{TN+FP} \quad (7)$$

3. *False Positive Rate* $$FPR = \frac{FP}{FP+TN} \quad (8)$$

4. *False Negative Rate* $$FNR = \frac{FN}{TP+FN} \quad (9)$$

5. *Percentage of Wrong Classifications*

$$PWC = 100 \times \frac{FN+FP}{TP+FN+FP+TN} \quad (10)$$

6. *Precision* $$Prec = \frac{TP}{TP+FP} \quad (11)$$

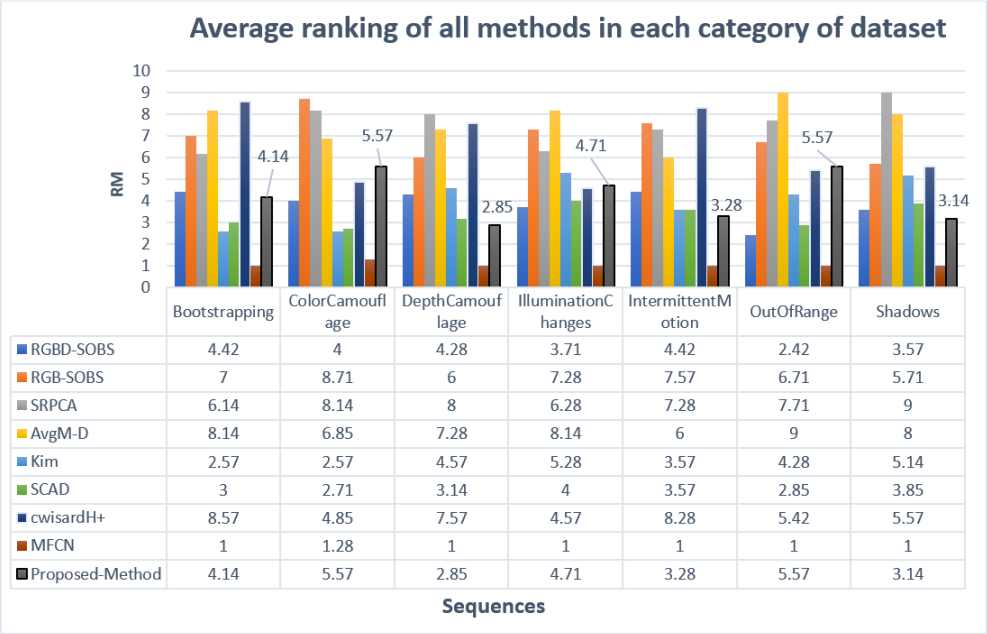
7. *F-Measure* $$F = \frac{2 \times Prec \times Rec}{Prec + Rec} \quad (12)$$

In addition, for more precise comparison the evaluation method in [33] is used to calculate the average ranking of method (RM) which combine the performance of each method across different metrics in each sequence.

Formally, let us denote the $N_m$as number of metrics in sequence *sq* and ranking of the *ith* method for the metric m as $rank_i(m, sq)$. The average ranking of method *i* is given as:

$$RM_i = \frac{1}{N_m}\sum_{i=1}^{m} rank_i(m, sq) \quad (13)$$

According to the figure 7, the MFCN method could detect the moving objects accurately in almost all categories and achieve the best results in all average results. However, this method requires many hours for training process of this dataset which is extensive process compared to our method which requires only a few seconds to create the model. After MFCN, the proposed method could achieve the best results in *DepthCam*, *Intermittent motion* and *shadows* sequences. Accurate detection in *DepthCam* sequences shows the method could manage the weakness of depth sensors (absence of depth value, similar depth and sensor error in the depth value) by using color frames. On the other hand, it also managed the appearance of shadows by using L*a*b* color and shadow detection method. The adaptive blind update significantly aided the method to manage intermittent motion sequences which normally is based on appearance of a new object or removing an object from the background. The weakest detection of the proposed method and most of other methods belongs to *Bootstrapping* sequences. The reason for this is that the moving object is included in the background from the beginning and consequently it will be added to the background model. Therefore, the moving object will be assumed as a background. In some bootstrapping sequences the ground truth starts from sequence 1 which is impossible for our method to detect moving object (person) as the background model has not yet created. Therefore, we have added histograms of oriented gradient [34] to help the detection of people in the beginning of each sequence.



	Bootstrapping	ColorCamouflage	DepthCamouflage	IlluminationChanges	IntermittentMotion	OutOfRange	Shadows
RGBD-SOBS	4.42	4	4.28	3.71	4.42	2.42	3.57
RGB-SOBS	7	8.71	6	7.28	7.57	6.71	5.71
SRPCA	6.14	8.14	8	6.28	7.28	7.71	9
AvgM-D	8.14	6.85	7.28	8.14	6	9	8
Kim	2.57	2.57	4.57	5.28	3.57	4.28	5.14
SCAD	3	2.71	3.14	4	3.57	2.85	3.85
cwisardH+	8.57	4.85	7.57	4.57	8.28	5.42	5.57
MFCN	1	1.28	1	1	1	1	1
Proposed-Method	4.14	5.57	2.85	4.71	3.28	5.57	3.14

Fig. 7. Average ranking of all existing methods in each category of dataset. A lower amount shows better performance in the category.

On other sequences the results are acceptable compared to the other methods which shows the stability of the algorithm in all sequences. In other word, it is proven that it is able to detect most moving objects in these challenging scenarios and does not completely fail in any scenario.

C. Real-time Experiment:

The proposed system has been tested in a live application with the processing time of approximately 12 fps for a 640x480 video. The computational cost of the algorithm is calculated as the mean rate of the processing time of the algorithm. The test was performed on a computer with an Intel(R) Core(TM) i7-6700HQ CPU @2.6 GHz and 8 GB RAM along with Microsoft Kinect v2 sensor.

It is worth mentioning that the proposed algorithm used during these tests has been implemented in C++ and OpenCv library [35] without any specific code optimisation as the aim of this experiment is to show that the proposed algorithm can be successfully run at real-time frame rates and therefore no effort has been made to optimise the code/set-up.

V. Conclusion

In this paper, a moving object detection using adaptive blind updating is proposed. The main contribution of this paper is adding adaptive blind update method, more complex segmentation, proposed bootstrapping detection and segmentation, proposed shadow detection method based on L*a*b* color space and complete evaluation of the proposed method. By tracking the moving object, the frequently of blind update will be changed according to the speed of moving object. This strategy will help the scene model to adapt to the complex scenarios such as environment changes, illumination changes and shadows as well as detecting the fast, slow and stationary objects while reducing the ghost phenomenon.

A simple tracking method has been used in this work to reduce the computational costs.

Experimental results show that the proposed method can significantly improve the accuracy of the algorithm when stationary objects are existing. On the other hand, on fast moving sequences, the algorithm achieved a slightly improved or equal results to the original method. In general, the main advantages of the proposed method is to improve segmentation accuracy in stationary moving objects, bootstrapping, shadow and depth camouflage scenarios. Overall the proposed method proved that is more consistent in all scenario. The main disadvantage of the proposed method is that the system has a higher computational cost compared to the original method. However, still this method can be applied in live applications. The best performance of this method is only achievable when one moving object exists. In the future, more complex decision-making system can be developed to deal with more than one moving objects.

References

[1] N. Dorudian, S. Lauria, and S. Swift, “Nonparametric background modelling and segmentation to detect Micro Air Vehicles (MAV) using RGB-D Sensor,” *Int. J. Micro Air Veh. to be published http//bura.brunel.ac.uk/handle/2438/17092*, 2018.

[2] C. Cuevas, R. Martínez, and N. García, “Detection of stationary foreground objects : A survey,” *Comput. Vis. Image Underst.*, vol. 152, pp. 41–57, 2016.

[3] G. Moyà-Alcover, A. Elgammal, A. Jaume-i-Capó, and J. Varona, “Modeling depth for nonparametric foreground segmentation using RGBD devices,” *Pattern Recognit. Lett.*, vol. 96, no. C, pp. 76–85, Sep. 2017.

[4] O. Barnich and M. Van Droogenbroeck, “ViBe : A Universal Background Subtraction Algorithm for Video Sequences,” *IEEE Trans. Image Process. vol. 20, no. 6*, vol. 20, no. JULY 2011, pp. 1709–1724.

[5] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” *Proc. 1999 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Cat No PR00149,IEEE Comput. Soc*, vol. 2, no. c, pp. 246–252, 1999.

[6] A. Benamara, S. Miguet, and M. Scuturici, “Real-Time Multi-object Tracking with Occlusion and Stationary Objects Handling,” vol. 2, pp. 136–145, 2016.

[7] H. Wang and L. Shi, “Foreground Model for Background Subtraction with Blind Updating,” pp. 74–78, 2016.

[8] Z. Zivkovic, “Improved adaptive Gaussian mixture model for background subtraction,” *Proc. 17th Int. Conf. Pattern Recognition, 2004. ICPR 2004.*, vol. 2, pp. 28–31, 2004.

[9] R. Martínez, C. Cuevas, D. Berjón, and N. García, “Detection of static moving objects using multiple nonparametric background models,” pp. 4–5, 2015.

[10] J. Leens, S. Piérard, O. Barnich, M. Van Droogenbroeck, and J.-M. Wagner, “Combining Color, Depth, and Motion for Video Segmentation,” Springer Berlin Heidelberg, 2009, pp. 104–113.

[11] S. Pierard and M. Van Droogenbroeck, “Techniques to improve the foreground segmentation with a 3D camera and a color camera,” *20th Annu. Work. Circuits, Syst. Signal Process.*, pp. 247–250, 2009.

[12] S. Ottonelli, P. Spagnolo, P. L. Mazzeo, and M. Leo, “Improved video segmentation with color and depth using a stereo camera,” *2013 IEEE Int. Conf. Ind. Technol.*, pp. 1134–1139, Feb. 2012.

[13] C. Sun, Y. Wang, and M. Sheu, “Fast Motion Object

Detection Algorithm Using Complementary Depth Image on an RGB-D Camera," *IEEE Sens. J.*, vol. 17, no. 17, pp. 5728–5734, 2017.

[14] S. Lee *et al.*, "Detection of a Suicide by Hanging Based on a 3-D Image Analysis," *IEEE Sens. J.*, vol. 14, no. 9, pp. 2934–2935, 2014.

[15] J. Hernández-Aceituno, R. Arnay, J. Toledo, and L. Acosta, "Using Kinect on an Autonomous Vehicle for Outdoors Obstacle Detection," *IEEE Sens. J.*, vol. 16, no. 10, pp. 3603–3610, May 2016.

[16] S. Javed, T. Bouwmans, M. Sultana, and S. K. Jung, "Moving Object Detection on RGB-D Videos Using Graph Regularized Spatiotemporal RPCA," Springer, Cham, 2017, pp. 230–241.

[17] M. Braham, A. Lejeune, and M. Van Droogenbroeck, "A physically motivated pixel-based model for background subtraction in 3D images," *2014 Int. Conf. 3D Imaging, IC3D 2014 - Proc.*, 2015.

[18] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

[19] L. Yang, L. Zhang, H. Dong, A. Alelaiwi, and A. El Saddik, "Evaluating and Improving the Depth Accuracy of Kinect for Windows v2," *IEEE Sens. J.*, vol. 15, no. 8, pp. 4275–4285, 2015.

[20] A. M. Pinto, P. Costa, A. P. Moreira, L. F. Rocha, E. Moreira, and G. Veiga, "Evaluation of depth sensors for robotic applications," *Proc. - 2015 IEEE Int. Conf. Auton. Robot Syst. Compet. ICARSC 2015*, pp. 139–143, 2015.

[21] R. S. (Richard S. Hunter and R. W. (Richard W. Harold, *The measurement of appearance*. Wiley, 1987.

[22] M. Camplani, C. R. Del Blanco, L. Salgado, F. Jaureguizar, and N. García, "Advanced background modeling with RGB-D sensors through classifiers combination and inter-frame foreground prediction," *Mach. Vis. Appl.*, vol. 25, no. 5, pp. 1197–1210, 2014.

[23] M. Camplani and L. Salgado, "Background foreground segmentation with RGB-D Kinect data: An efficient combination of classifiers," *J. Vis. Commun. Image Represent.*, vol. 25, no. 1, pp. 122–136, 2014.

[24] "M. Camplani, L. Maddalena, G. Moyà Alcover, A. Petrosino, L. Salgado,A Benchmarking Framework for Background Subtraction in RGBD videos, in S. Battiato, G. Gallo, G.M. Farinella, M. Leo (Eds), New Trends in Image Analysis and Processing-ICIAP 2017 Worksho," *Lect. Notes Comput. Sci. Springer, 2017*.

[25] L. Li, W. Huang, I. Y.-H. Gu, and Q. Tian, "Statistical Modeling of Complex Backgrounds for Foreground Object Detection," *IEEE Trans. Image Process.*, vol. 13, no. 11, pp. 1459–1472, Nov. 2004.

[26] R. Real, J. M. Vargas, and R. Olmstead, "The Probabilistic Basis of Jaccard's Index of Similarity," *Syst. Biol.*, vol. 45, no. 3, pp. 380–385, Sep. 1996.

[27] L. Maddalena and A. Petrosino, "Exploiting Color and Depth for Background Subtraction," Springer, Cham, 2017, pp. 254–265.

[28] L. Maddalena and A. Petrosino, "The SOBS algorithm: What are the limits?," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 21–26.

[29] T. Minematsu, A. Shimada, H. Uchiyama, and R. Taniguchi, "Simple Combination of Appearance and Depth for Foreground Segmentation," Springer, Cham, 2017, pp. 266–277.

[30] M. De Gregorio and M. Giordano, "CwisarDH+ : Background Detection in RGBD Videos by Learning of Weightless Neural Networks," Springer, Cham, 2017, pp. 242–253.

[31] D. Zeng and M. Zhu, "Background Subtraction Using Multiscale Fully Convolutional Network," *IEEE Access*, vol. 6, pp. 16010–16021, 2018.

[32] L. Maddalena, "Results of the SBM-RGBD Challenge," 2018. [Online]. Available: http://rgbd2017.na.icar.cnr.it/SBM-RGBDchallengeResults.html. [Accessed: 01-Aug-2018].

[33] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.net: A new change detection benchmark dataset," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 2012, pp. 1–8.

[34] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893.

[35] I. Intel Corporation, Willow Garage, "Open Source Computer Vision Library," 2018. [Online]. Available: http://opencv.org.